



CHAPTER 5: USING PSCP TO TRANSFER FILES SECURELY

PSCP, the PuTTY Secure Copy client, is a tool for transferring files securely between computers using an SSH connection.

If you have an SSH-2 server, you might prefer PSFTP (see [chapter 6](#)) for interactive use. PSFTP does not in general work with SSH-1 servers, however.

5.1 STARTING PSCP

PSCP is a command line application. This means that you cannot just double-click on its icon to run it and instead you have to bring up a console window. With Windows 95, 98, and ME, this is called an 'MS-DOS Prompt' and with Windows NT, 2000, and XP, it is called a 'Command Prompt'. It should be available from the Programs section of your Start Menu.

To start PSCP it will need either to be on your `PATH` or in your current directory. To add the directory containing PSCP to your `PATH` environment variable, type into the console window:

```
set PATH=C:\path\to\putty\directory;%PATH%
```

This will only work for the lifetime of that particular console window. To set your `PATH` more permanently on Windows NT, 2000, and XP, use the Environment tab of the System Control Panel. On Windows 95, 98, and ME, you will need to edit your `AUTOEXEC.BAT` to include a `set` command like the one above.

5.2 PSCP USAGE

Once you've got a console window to type into, you can just type `pscp` on its own to bring up a usage message. This tells you the version of PSCP you're using, and gives you a brief summary of how to use PSCP:

```
Z:\owendadmin>pscp
PuTTY Secure Copy client
Release 0.68
Usage: pscp [options] [user@]host:source target
       pscp [options] source [source...] [user@]host:target
       pscp [options] -ls [user@]host:filespec
Options:
  -V          print version information and exit
  -pgpfp      print PGP key fingerprints and exit
  -p          preserve file attributes
  -q          quiet, don't show statistics
  -r          copy directories recursively
  -v          show verbose messages
  -load sessname Load settings from saved session
  -P port     connect to specified port
  -l user     connect with specified username
  -pw passw   login with specified password
  -1 -2       force use of particular SSH protocol version
  -4 -6       force use of IPv4 or IPv6
  -C          enable compression
  -i key      private key file for user authentication
  -noagent    disable use of Pageant
  -agent      enable use of Pageant
  -hostkey aa:bb:cc:...
              manually specify a host key (may be repeated)
  -batch      disable all interactive prompts
  -proxycmd command
              use 'command' as local proxy
  -unsafe     allow server-side wildcards (DANGEROUS)
  -sftp       force use of SFTP protocol
  -scp        force use of SCP protocol
  -sshlog file
```

```
-ssnlog file  
-sshrawlog file  
log protocol details to a file
```

(PSCP's interface is much like the Unix `scp` command, if you're familiar with that.)

5.2.1 THE BASICS

To receive (a) file(s) from a remote server:

```
pscp [options] [user@]host:source target
```

So to copy the file `/etc/hosts` from the server `example.com` as user `fred` to the file `c:\temp\example-hosts.txt`, you would type:

```
pscp fred@example.com:/etc/hosts c:\temp\example-hosts.txt
```

To send (a) file(s) to a remote server:

```
pscp [options] source [source...] [user@]host:target
```

So to copy the local file `c:\documents\foo.txt` to the server `example.com` as user `fred` to the file `/tmp/foo` you would type:

```
pscp c:\documents\foo.txt fred@example.com:/tmp/foo
```

You can use wildcards to transfer multiple files in either direction, like this:

```
pscp c:\documents\*.doc fred@example.com:docfiles  
pscp fred@example.com:source/*.c c:\source
```

However, in the second case (using a wildcard for multiple remote files) you may see a warning saying something like 'warning: remote host tried to write to a file called `terminal.c` when we requested a file called `*.c`'. If this is a wildcard, consider upgrading to SSH-2 or using the `-unsafe` option. Renaming of this file has been disallowed'.

This is due to a fundamental insecurity in the old-style SCP protocol: the client sends the wildcard string (`*.c`) to the server, and the server sends back a sequence of file names that match the wildcard pattern. However, there is nothing to stop the server sending back a *different* pattern and writing over one of your other files: if you request `*.c`, the server might send back the file name `AUTOEXEC.BAT` and install a virus for you. Since the wildcard matching rules are decided by the server, the client cannot reliably verify that the filenames sent back match the pattern.

PSCP will attempt to use the newer SFTP protocol (part of SSH-2) where possible, which does not suffer from this security flaw. If you are talking to an SSH-2 server which supports SFTP, you will never see this warning. (You can force use of the SFTP protocol, if available, with `-sftp` - see [section 5.2.2.6.](#))

If you really need to use a server-side wildcard with an SSH-1 server, you can use the `-unsafe` command line option with PSCP:

```
pscp -unsafe fred@example.com:source/*.c c:\source
```

This will suppress the warning message and the file transfer will happen. However, you should be aware that by using this option you are giving the server the ability to write to *any* file in the target directory, so you should only use this option if you trust the server administrator not to be malicious (and not to let the server machine be cracked by malicious people). Alternatively, do any such download in a newly created empty directory. (Even in 'unsafe' mode, PSCP will still protect you against the server trying to get out of that directory using pathnames including `..`.)

5.2.1.1 USER

The login name on the remote server. If this is omitted, and `host` is a PuTTY saved session, PSCP will use any username specified by that saved session. Otherwise, PSCP will attempt to use the local Windows username.

5.2.1.2 HOST

The name of the remote server, or the name of an existing PuTTY saved session. In the latter case, the session's settings for hostname, port number, cipher type and username will be used.

5.2.1.3 SOURCE

One or more source files. Wildcards are allowed. The syntax of wildcards depends on the system to which they apply, so if you are copying *from* a Windows system *to* a UNIX system, you should use Windows wildcard syntax (e.g. `*.*`), but if you are copying *from* a UNIX system *to* a Windows system, you would use the wildcard syntax allowed by your UNIX shell (e.g. `*`).

If the source is a remote server and you do not specify a full pathname (in UNIX, a pathname beginning with a `/` (slash) character), what you specify as a source will be interpreted relative to your home directory on the remote server.

5.2.1.4 TARGET

The filename or directory to put the file(s). When copying from a remote server to a local host, you may wish simply to place the file(s) in the current directory. To do this, you should specify a target of `..`. For example:

```
pscp fred@example.com:/home/tom/.emacs ..
```

...would copy `/home/tom/.emacs` on the remote server to the current directory.

As with the `source` parameter, if the target is on a remote server and is not a full path name, it is interpreted relative to your home directory on the remote server.

5.2.2 OPTIONS

PSCP accepts all the general command line options supported by the PuTTY tools, except the ones which make no sense in a file transfer utility. See [section 3.8.3](#) for a description of these options. (The ones not supported by PSCP are clearly marked.)

PSCP also supports some of its own options. The following sections describe PSCP's specific command-line options.

5.2.2.1 -LS LIST REMOTE FILES

If the `-ls` option is given, no files are transferred; instead, remote files are listed. Only a hostname specification and optional remote file specification need be given. For example:

```
pscp -ls fred@example.com:dir1
```

The SCP protocol does not contain within itself a means of listing files. If SCP is in use, this option therefore assumes that the server responds appropriately to the command `ls -la`; this may not work with all servers.

If SFTP is in use, this option should work with all servers.

5.2.2.2 -P PRESERVE FILE ATTRIBUTES

By default, files copied with PSCP are timestamped with the date and time they were copied. The `-p` option preserves the original timestamp on copied files.

5.2.2.3 -Q QUIET, DON'T SHOW STATISTICS

By default, PSCP displays a meter displaying the progress of the current transfer:

```
mibs.tar          | 168 kB | 84.0 kB/s | ETA: 00:00:13 | 13%
```

The fields in this display are (from left to right), filename, size (in kilobytes) of file transferred so far, estimate of how fast the file is being transferred (in kilobytes per second), estimated time that the transfer will be complete, and percentage of the file so far transferred. The `-q` option to PSCP suppresses the printing of these statistics.

5.2.2.4 -R COPIES DIRECTORIES RECURSIVELY

By default, PSCP will only copy files. Any directories you specify to copy will be skipped, as will their contents. The `-r` option tells PSCP to descend into any directories you specify, and to copy them and their contents. This allows you to use PSCP to transfer whole directory structures between machines.

5.2.2.5 -BATCH AVOID INTERACTIVE PROMPTS

If you use the `-batch` option, PSCP will never give an interactive prompt while establishing the connection. If the server's host key is invalid, for example (see [section 2.2](#)), then the connection will simply be abandoned instead of asking you what to do next.

This may help PSCP's behaviour when it is used in automated scripts: using `-batch`, if something goes wrong at connection time, the batch job will fail rather than hang.

5.2.2.6 -SFTP, -SCP FORCE USE OF PARTICULAR PROTOCOL

As mentioned in [section 5.2.1](#), there are two different file transfer protocols in use with SSH. Despite its name, PSCP (like many other ostensible `scp` clients) can use either of these protocols.

The older SCP protocol does not have a written specification and leaves a lot of detail to the server platform. Wildcards are expanded on the server. The simple design means that any wildcard specification supported by the server platform (such as brace expansion) can be used, but also leads to interoperability issues such as with filename quoting (for instance, where filenames contain spaces), and also the security issue described in [section 5.2.1](#).

The newer SFTP protocol, which is usually associated with SSH-2 servers, is specified in a more platform independent way, and leaves issues such as wildcard syntax up to the client. (PuTTY's SFTP wildcard syntax is described in [section 6.2.2](#).) This makes it more consistent across platforms, more suitable for scripting and automation, and avoids security issues with wildcard matching.

Normally PSCP will attempt to use the SFTP protocol, and only fall back to the SCP protocol if SFTP is not available on the server.

The `-scp` option forces PSCP to use the SCP protocol or quit.

The `-sftp` option forces PSCP to use the SFTP protocol or quit. When this option is specified, PSCP looks harder for an SFTP server, which may allow use of SFTP with SSH-1 depending on server setup.

5.2.3 RETURN VALUE

PSCP returns an `ERRORLEVEL` of zero (success) only if the files were correctly transferred. You can test for this in a batch file, using code such as this:

```
pscp file*.* user@hostname:
if errorlevel 1 echo There was an error
```

5.2.4 USING PUBLIC KEY AUTHENTICATION WITH PSCP

Like PuTTY, PSCP can authenticate using a public key instead of a password. There are three ways you can do this.

Firstly, PSCP can use PuTTY saved sessions in place of hostnames (see [section 5.2.1.2](#)). So you would do this:

- Run PuTTY, and create a PuTTY saved session (see [section 4.1.2](#)) which specifies your private key file (see [section 4.22.8](#)). You will probably also want to specify a username to log in as (see [section 4.14.1](#)).
- In PSCP, you can now use the name of the session instead of a hostname: type `pscp sessionname:file localfile`, where `sessionname` is replaced by the name of your saved session.

Secondly, you can supply the name of a private key file on the command line, with the `-i` option. See [section 3.8.3.18](#) for more information.

Thirdly, PSCP will attempt to authenticate using Pageant if Pageant is running (see [chapter 9](#)). So you would do this:

- Ensure Pageant is running, and has your private key stored in it.
- Specify a user and host name to PSCP as normal. PSCP will automatically detect Pageant and try to use the keys within it.

For more general information on public-key authentication, see [chapter 8](#).

If you want to provide feedback on this manual or on the PuTTY tools themselves, see the [Feedback page](#).

[PuTTY release 0.68]



TRY

PRIVX IS IN YOUR BROWSER!

Start free, then pay monthly when you grow. Free for up to 20 hosts - no credit card required

TRY PRIVX NOW! 

What to read next:

- **REDUCE SECURE SHELL RISK. GET TO KNOW THE NIST 7966.**



The NISTIR 7966 guideline from the Computer Security Division of NIST is a direct call to action for organizations regardless of industry and is a mandate for the US Federal government.

[Download now](#)

- **ISACA PRACTITIONER GUIDE FOR SSH**



With contributions from practitioners, specialists and SSH.COM experts, the ISACA "SSH: Practitioner Considerations" guide is vital best practice from the compliance and audit community.

[Download now](#)

NEAREST SSH.COM OFFICE

SSH Communications Security, Inc.
460 Totten Pond Road #460
Waltham, MA 02140
USA
[781-247-2100](tel:781-247-2100)
[Contact us](#)

QUICK LINKS

[Get a demo](#)
[Blog](#)
[PrivX](#)
[Universal SSH Key Manager software](#)
[CryptoAuditor](#)
[Tectia](#)
[Support](#)
[About us](#)
[Contact](#)
[Home](#)
[Auf Deutsch](#)
[En español](#)
[In het Nederlands](#)
[En français](#)

Copyright ©2019 SSH Communications Security, Inc. All Rights Reserved. [See Data Privacy Policy, Website Terms of Use, and Standard Terms and Conditions EULAs](#)

